User's Manual for ASPIC: A Stock-Production Model Incorporating Covariates (ver. 5) And Auxiliary Programs

Michael H. Prager — Prager Consulting
Portland, Oregon, USA
www.mhprager.com

Last revised April 17, 2011

Preface

This user's manual describes Version 5.0 of ASPIC, a computer program to estimate parameters of a non-equilibrium surplus-production model from fisheries data. Several utility programs (ASPICP, FTEST, AGRAPH) are also described. Their purposes are making projections, comparing models, and quickly making graphs from ASPIC and ASPICP output files. The programs together are referred to here as the ASPIC Suite.

The major change from previous versions of ASPIC is the ability to fit the Pella–Tomlinson (generalized) production model, with the Fox exponential yield model included as a special case. The Schaefer (logistic) production model, the main component of earlier versions, is still part of ASPIC, and because many of its computations can be done analytically rather than numerically, it will be found quicker, and its solutions may be more stable.

The ASPIC Suite is not commercial software, and the programs are not warranted in any way, either by the author or by any of the sponsors who made development possible. The software was developed for use in the author's research, and it is used regularly for stock assessment and education. Distribution to fellow scientists in made in a cooperative spirit. *The software is intended as a set of research tools, and those who use them do so at their own risk.* ASPIC has been used on thousands of real and simulated data sets, and all supplied programs are believed to be substantially correct. The author appreciates receiving advice of suspected flaws, and he attempts to correct errors promptly.

By no means is ASPIC the final word in production modeling. It is intended as a reasonably flexible program that can serve as a basis for further innovation.

Formal description of the theory behind ASPIC is given in Prager (1994). Further references are given in the bibliography. The author requests that this manual and Prager (1994) be cited in any report or published article that uses ASPIC.

Those who have used version 3.x of ASPIC and who now are presented with version 5.x might ask what happened to version 4.x. The answer is simple: 4.x were test versions. It seemed more logical to release the new version as 5.0, rather than some number in the middle of the 4.x series.

Many colleagues have given valuable technical suggestions or assistance while ASPIC was being written and as it has been revised through the years. I thank S. Cadrin, R. Deriso, K. Hiramatsu, J. Hoenig, R. Methot, C. Porch, J. Powers, A. Punt, V. Restrepo, G. Scott, K. Shertzer, P. Tomlinson, D. Vaughan, E. Williams, and the many fishery scientists who have sent data sets to illustrate their applications and questions.

Development of ASPIC and related research has been supported by the Southeast Fisheries Science Center of the U.S. National Marine Fisheries Service and by the author's private efforts. This software is distributed to interested scientists free of charge. No individual or group is authorized to charge for it or distribute it as part of any commercial product.

This manual is based on National Marine Fisheries Service Miami Laboratory Document MIA–92/93–55, which was later revised into Beaufort Laboratory Document BL–2004–01, dated February, 2004.

Typographical conventions

In this manual, user commands, file names, and items in input files are displayed in a `monospaced font`. Some important sections ❖ are marked by a symbol in the margin, as here; attention to such material is especially important to obtaining good results from ASPIC. Material new in this version of the program is marked by a different ✺ marginal symbol, as here.

Michael H. Prager
Beaufort, North Carolina
January, 2004, and May, 2010

Contents

Preface 2

Typographical conventions 2

Contents 3

List of Tables 4

1 Introduction 5

2 New in version 5.0 5

2.1 Major changes 5

2.2 Using new features 6

2.2.1 Execution speed 6

2.2.2 Developmental features 7

3 Installation and Interaction 7

3.1 Compatibility 7

3.2 Installation 7

3.3 Interface 7

3.3.1 ASPIC and ASPICP 7

3.3.2 Auxiliary programs 8

3.3.3 NMFS Toolbox 8

4 Overview of ASPIC 8

4.1 Data requirements 8

4.2 Program limits 8

4.3 Program modes 8

4.4 Several data series 9

4.5 Objective function 9

4.5.1 Penalty for initial biomass 9

4.5.2 Conditioning on yield 9

4.6 Bootstrapping 10

4.7 Input and output files 10

4.7.1 Input file 10

4.7.2 Editing input files 10

4.7.3 Output files 10

4.8 Starting ASPIC 12

5 Auxiliary Programs 12

5.1 Overview of ASPICP 12

5.2 Overview of FTEST 12

5.3 Overview of AGRAPH 13

6 ASPIC Input File Specification 13

6.1 Generating a sample input file 13

6.2 General format guidelines 13

6.3 The ASPIC input file, line by line 13

6.4 Starting guesses and bounds 18

6.5 Questions about data series 18

6.5.1 Missing values and zeroes 18

6.5.2 Allocating yield 19

6.5.3 Caution 20

7 Advanced Options for ASPIC 20

8 ASPICP Input File Specification 20

8.1 Line by line 20

8.2 Sample input file 21

9 Interpretation of ASPIC Results 21

9.1 Precision of parameter estimates 21

9.2 Using several data series 22

9.2.1 Catchability over time 22

9.3 Projections 23

9.4 Estimation difficulties 23

10 Program Changes 24

11 Source Code 25

References 25

List of Tables

1 Data-series types and codes 10
2 Files used by the ASPIC suite 11
– Model shape codes 14
– Conditioning codes 14
– Objective function codes 14
– Model shape settings 14
3 Missing values and zeroes 19

1 Introduction

This user's manual describes Version 5.0 of ASPIC, a computer program to estimate parameters of a non-equilibrium surplus-production model from fisheries data. Several utility programs (ASPICP, FTEST, AGRAPH) are also described. Their purposes include making projections, comparing models, and quickly making graphs from ASPIC and ASPICP output files. The programs together are referred to here as the ASPIC Suite.

The surplus-production model has a long history in fishery science and has repeatedly proven useful in management of fish stocks. The appeal of production models is in large part due to their conceptual and computational simplicity. Despite that simplicity, production models incorporate an implicit recruitment function, and thus can be used for studies of sustainability. Production models have also been found especially useful in stock assessments when the age-structure of the catch cannot be estimated.

Many early treatments of surplus-production models assumed that the yield taken each year could be considered the equilibrium yield (e. g., Fox 1975). However using that "equilibrium assumption" tends to overestimate MSY when used to assess a declining stock, and it has been found problematic by several studies (Mohn 1980; Williams and Prager 2002). The assumption was a computational convenience that is no longer needed, and ASPIC does not use it.

✺ Earlier versions of ASPIC could fit only the logistic production model (Schaefer 1954, 1957; Pella 1967), in which the production curve (curve of surplus production vs. biomass) is symmetrical around MSY. Version 5.x also fits the generalized model of Pella and Tomlinson (1969) in the revised parameterization of Fletcher (1978).

ASPIC incorporates several extensions to classical stock-production models. One extension is that ASPIC can fit data from up to 10 data series. These may be catch–effort series (from different gears or different periods of time), catch–abundance-index series, biomass indices, or biomass estimates made independently of the production model. This feature is described in §4.4. A second major extension is the use of bootstrapping for bias correction and construction of approximate nonparametric confidence intervals. A third extension is that ASPIC can fit a model under the assumption that yield in each year is known more precisely than fishing effort or relative abundance; in other words, fitting can be statistically conditioned on yield, rather than on fishing effort or relative abundance.

The theory behind ASPIC and several worked examples were first presented in working documents of the International Commission for the Conservation of Atlantic Tunas (ICCAT) by Prager (1992a,b). Those reference have been superseded by the more formal and complete treatment of Prager (1994). The model and its extensions are also described in Quinn and Deriso (1999) and Haddon (2001). The basic theory of production models is of course also described in many other texts, including Hilborn and Walters (1992), and is the subject of a recent FAO publication (Punt and Hilborn 1996).

The ASPIC computer program as described here has been used by several assessment groups and in many studies, including Prager et al. (1996), Prager and Goodyear (2001), Prager (2002), Shertzer and Prager (2002), and Williams and Prager (2002). In the course of those studies, the program has been exercised on over 100,000 sets of simulated data. The resulting experience has been used to improve the program's reliability.

Versions of ASPIC before 5.0 are no longer maintained by the author. They are still available for those who wish to duplicate old analyses exactly. However, the author urges analysts to use current versions for all new work. ❖

2 New in version 5.0 ❖

This section gives an overview of changes introduced between ASPIC 3.x and ASPIC 5.x. Although this section will be of most interest to users of previous versions, new users should also review it briefly for information on running ASPIC 5.0 productively.

2.1 Major changes ✺

Generalized production model. Earlier versions of ASPIC could fit only the logistic form of the production model (Graham 1935; Schaefer 1954, 1957; Pella 1967; Prager 1994). As well as that form, ASPIC 5.0 can fit the generalized production model (Pella and Tomlinson 1969; Fletcher 1978) in one of three ways: by direct optimization, by a grid of fits on the model shape, or with fixed model shape to implement the Fox (1970) model or other pre-determined shape.

Parameterization change. The generalized model requires parameterization in terms of MSY and carrying capacity K rather than MSY and intrinsic rate of increase r. This occurs because when the exponent n in the generalized model is in the region $n \leq 1$, then $r = \infty$. As a result, ASPIC 5.0 requires a starting guess for K, not for r.

Starting biomass parameterization. A parameter estimated by ASPIC is the biomass in the first year of the analysis. In previous versions, this was expressed (both in the input file and in ASPIC reports) as a ratio to the biomass providing MSY; i.e., as B_1/B_{MSY}. In version 5.0, it is expressed as a ratio to the carrying capacity, i.e., as B_1/K. This change is required because in the generalized model, B_{MSY} is no longer a fixed proportion of K. The change reduces correlation between estimates of the starting biomass ratio and of B_{MSY}.

Conditioning options. Option names for conditioning on yield or effort have been revised to indicate conditioning, rather than residuals. The new specifications are given in §6.3.

Fitting criteria. An additional objective function, least absolute values (LAV), is available. It is recommended that this robust objective function be used only in conjunction with a regular least squares fit, because the optimizer has a more difficult task in finding the best minimum of LAV fits. Nonetheless, LAV can be valuable where one or more data observations are markedly disjoint from the rest. For guidelines on appropriate use of LAV, please consult the statistical literature.

Bounds on catchability coefficient. To improve convergence, estimates of q (catchability) are now bounded to a geometric range around the user's starting guess. The bounds are determined internally by ASPIC and are not under the user's control. If the starting guess for q is severely wrong, the estimate may hit a bound, and the ASPIC report will indicate whether the starting guess was too low or too high. If that happens, the user should revise the starting guess of q accordingly and rerun the analysis.

Restarts during optimization. Previous versions of ASPIC required the optimizer to return to the same solution 3 times in a row to indicate convergence. That worked well on most data sets, but was not always sufficient. The number of identical returns is now specified in the input file. The recommended default is 8. This can improve stability of the fit on some poor data sets. The value is specified on line 8 of the input file; see §6.3 on page 16.

Time steps. The generalized model is implemented by numerical integration that approximates a continuous–time solution. The number of time steps per year is specified on line 9 of the input file; see §6.3 on page 16.

Length of series. Time series in the input file may now be up to 250 years long.

Updated ASPICP. An updated version of the projection program ASPICP is compatible with analyses from ASPIC 5.0. It is also backwardly compatible with ASPIC 3.x and 4.x. Starting with version 3.15 of ASPICP, the projection (output) filename is derived from the input filename, rather than being specified by the user in the input file.

Windows installer. This release is distributed as a self-installing binary file for Windows. Versions for other operating systems may be available on request.

Drag-and-drop versions. ASPIC has always been a non-interactive program that reads from and writes to ASCII files. This release includes alternative versions that accept drag-and-drop of input files and that display their output in scrolling windows. The original command-line versions are also supplied.

2.2 Using new features

When using the new features of ASPIC 5.0, please consider the following—

2.2.1 Execution speed

Fitting the generalized model is done with a numerical solution of the catch equation and is thus slower than fitting the logistic model (which has an analytical solution). Execution will be especially slow in any the following cases:

1. Poor agreement between model and data
2. Analysis of more than one data series

3. Bootstrapping, especially in combination with #1 or #2

4. Extensive Monte Carlo trials in fitting

Program speed can be improved by reducing the number of time steps per year in the input file (see §6.3 on page 16). For the closest approximation to a continuous-time model, set this value to a large number; e.g., 80. For a fairly close approximation, use a number in the range 12–24 (recommended). For fastest operation set this to 2 steps per year.

❖ 2.2.2 Developmental features

The following are believed to work correctly but have not been tested extensively:

- Generalized estimation (in all forms) with more than one data series.
- Objective functions other than SSE.
- Projections of generalized bootstraps with ASPICP.
- The values of the AIC (Akaike Information Criterion) printed when fitting the generalized model.
- The *F*–statistic printed for comparing the logistic and generalized models may be incorrect when more than one data series is analyzed. More importantly, simulations suggest that such tests are of little value (Prager 2002).

❖ It is wise to repeat estimation with several different random-number seeds. If results cannot be duplicated within a few percent (usually less), a fitting failure is indicated, and such results should not be considered valid estimates.

❖ The author will appreciate receiving reports of successful or unsuccessful use of the features itemized above. He will attempt to fix all bugs promptly.

3 Installation and Interaction

3.1 Compatibility

The ASPIC suite is compatible with personal computers running Microsoft Windows 9x (including Windows 95, 98 and Me) or Windows NT (including Windows NT 4.0, 2000, and XP).[1]

[1] Use of tradenames does not imply endorsement by NMFS, NOAA, or the author.

ASPIC is written in standard Fortran 95 and is portable to other operating systems. Please consult the author if you would like to use ASPIC under operating systems other than Windows.

3.2 Installation

This version of ASPIC is available as a self-installing executable file for Windows. The installer performs the following tasks:

- Installs binary files for ASPIC, ASPICP, FTEST and AGRAPH to a location specified by the user
- Installs this User's manual and a Quick Reference Card to the `doc` subdirectory of the installation location
- Installs sample input and output files to the `samples` subdirectory of the installation location
- Adds the installation location to the user's PATH specification so that ASPIC and related programs can be executed from a command window open to any directory
- Adds a GINO environment variable pointing to the installation location, as required by a graphics support library used in AGRAPH. If a GINO environment variable already exists on the system, it is not modified.
- Adds shortcuts to the Windows Start menu and Desktop, including a command window opening in a user-specified working directory
- Adds an uninstaller to the installation location and adds ASPIC to the system's "Remove Programs" list

The ASPIC uninstaller removes all of the above. However, any files added by the user are not removed.

This User's Manual is supplied with all distributions of ASPIC as an Adobe PDF file named ASPICMAN.PDF. It may be distributed freely.

3.3 Interface ❖

3.3.1 Interface of ASPIC and ASPICP

The standard versions of ASPIC and ASPICP do not include graphical user interfaces. Instead, the programs are console-mode (character) programs that

read all input from and write all output to ASCII (text) files. The screen is used only for status messages.

For added ease of use, the ASPIC 5.0 installation includes versions of ASPIC and ASPICP that support drag and drop. (The executable files of the command-line versions are `aspic.exe` and `aspicp.exe`; of the drag-and-drop versions, `aspicw.exe` and `aspicpw.exe`.

3.3.2 Interface of auxiliary programs

The auxiliary program FTEST has a text-mode user interface, is interactive, and does not require an input file. Output is written to the screen.

The graphics program AGRAPH incorporates a standard Windows GUI, with output available to any Windows printer or to a graphics file (WMF or EPS). It can be executed from the command line, by drag and drop, or by starting the program from its shortcut.

3.3.3 NMFS Toolbox

The U.S. National Marine Fisheries Service (NMFS) has developed a "toolbox" of computer programs for stock assessment. The toolbox currently includes a graphical editor specifically for ASPIC input files and graphics that work with ASPIC output. It also includes many other stock–assessment and projection tools. For further information, contact Dr. Paul Rago at the NMFS laboratory in Woods Hole, Massachusetts.

4 Overview of ASPIC

4.1 Data requirements

Data needed by ASPIC are a series of observations on yield (catch in biomass) and one or more corresponding series of relative abundance. Data on fishing effort rate can be used instead of relative abundance, and when used are assumed to represent effective (standardized) effort. ASPIC assumes that the supplied abundance index is an unbiased index of the stock's abundance in biomass. If data on fishing effort are provided, ASPIC assumes that effort divided by yield forms an unbiased index of the stock's abundance.

In this User's Guide, the terms "catch" and "yield" are used interchangeably to mean total removals in biomass. Similarly, CPUE is used to mean relative abundance. The presumption is that the CPUE has been standardized before being used for modeling.

In addition to data, ASPIC requires starting guesses of its estimated parameters. Parameters directly estimated are K, the stock's maximum biomass or carrying capacity; MSY, the maximum sustainable yield; B_1/K, the ratio of the biomass at the beginning of the first year to K; and for each data series i, q_i, the catchability coefficient for that series. Description of the input file format, given in §6, includes suggestions for starting guesses.

4.2 Program limits

The array limits of ASPIC are as follows:

- Number of years of data: 250
- Number of data series: 10
- Number of bootstrap trials: 1,000

Any user with larger requirements is invited to contact the author.

4.3 Program modes

ASPIC has three modes of operation, here called "program modes."

- In FIT program mode, ASPIC fits the model and computes estimates of parameters and other quantities of management interest, including time trajectories of fishing intensity and stock biomass. Execution time is relatively short.
- In BOT program mode, ASPIC fits the model and computes bootstrapped confidence intervals on estimated quantities. Because computations are extensive, execution time in BOT mode is considerably longer than in FIT mode. For example, a bootstrap with 500 trials might take 200–500 times as long as a single fit.
- In IRF program mode, ASPIC conducts an iteratively reweighted fit when two or more data series are analyzed. Iterative reweighting of the data series (inverse-variance weighting) provides, under many circumstances, a maximum-likelihood solution.

The modes BOT and IRF cannot be combined. In other words, ASPIC cannot run a bootstrap on an iteratively reweighted fit.

A typical analysis might begin with FIT mode, including several runs to explore different model structures. If questions about series weighting are to be addressed, IRF mode might be used, instead, in the initial analysis. After model and data structure have been decided, BOT mode can be used to estimate the uncertainty in assessment results. ASPIC bootstrap runs do not incorporate iterative reweighting, and this will cause underestimation of variability when IRF mode has been used to develop a model structure.

IRF mode was developed in response to requests during assessment workshops, but it has not been much used by the author, and thus is less thoroughly tested than the other modes. Experience has shown that while series weights estimated in IRF mode may be statistically unbiased, they can be of high variance. (This is a characteristic of such weights generally, and is not specific to ASPIC.) For that reason, sensitivity to series weights should be examined whenever IRF program mode is used.

4.4 Fitting more than one data series

ASPIC can fit data on up to 10 simultaneous or serial fisheries (or biomass estimate series or biomass index series). Data series may be of several types (Table 1), but at least one series must be type CE (effort and yield) or type CC (CPUE and yield). When more than one series is analyzed, common estimates of B_1/K, MSY, and K are made, along with an estimate of q_i for each series. The interpretation of q_i depends on the type of data series to which it pertains.

A statistical weight w_i for each fishery is specified by the user in the input file. In summing the objective function, each squared residual from fishery i is multiplied by w_i. If the series have equal error variances, using weights of unity for each series provides a maximum-likelihood solution under the lognormal error structure assumed by ASPIC .

In FIT program mode, the program normalizes the user's w_i so that they sum to unity. In IRF mode, the program adjusts the weights interatively to provide nearly equal estimated variances. Weights are also adjusted so they sum to unity.

The computer time needed to obtain estimates generally increases as more data series are added. The increase is due both to addition of data and increased difficulty of optimization.

4.5 Objective function and penalty term

Parameters are estimated under the assumption that the errors in yield or effort are multiplicative with constant standard deviation. Thus the residuals are accumulated in logarithmic transform. The objective function Ω minimized, then, is

$$\Omega = \sum_{i=1}^{I} \sum_{j=1}^{N} w_i \left(\ln \frac{Y_{ij}}{\hat{Y}_{ij}} \right)^2 \qquad (1)$$

for residuals accumulated in yield (EFT optimization mode, §6.3), or a similar expression for residuals in effort (YLD optimization mode).

In equation (1), i indexes the data series, j the year, w is the series' statistical weight, Y_{ij} is the observed yield (or biomass index or estimates) from series i in year j, and $\hat{Y}_{ij}$ is the corresponding predicted value.

4.5.1 Penalty for initial biomass

A penalty term can be added to the objective function to discourage estimates in which the first year's biomass B_1 is greater than the carrying capacity K. This penalty can affect the estimates of other parameters, so when this term is used, the results should be compared to those obtained by setting the term to zero. The penalty term is described in more detail in Prager (1994), and its use is described in the section describing the input file format.

4.5.2 Conditioning on yield

ASPIC can consider yield known exactly and accumulate residuals in effort. Yield is usually observed more precisely than effort or the abundance index, and it is usually preferable on statistical grounds to compute residuals in the more imprecise quantity. Thus, conditioning on yield is recommended for most analyses. An additional advantage is that estimation of missing effort values is quite simple (and is included in this version of ASPIC). When conditioning on yield, an iterative solution of the catch equation is used, and computation is slower than when conditioning on effort.

Table 1. Codes for the eight types of data series allowed in ASPIC.

Code	Data type	When measured
CE	Fishing effort rate, catch (weight)	Effort rate: annual average Catch: annual total
CC	CPUE (weight-based), catch (weight)	CPUE: annual average Catch: annual total
B0	Estimate of biomass	Start of year
B1	Estimate of biomass	Annual average
B2	Estimate of biomass	End of year
I0	Index of biomass	Start of year
I1	Index of biomass	Annual average
I2	Index of biomass	End of year

4.6 Bootstrapped confidence intervals

In BOT mode, ASPIC uses bootstrapping to estimate bias-corrected confidence intervals on many quantities of interest. In doing this, estimated yields (if conditioning on effort; estimated efforts, if conditioning on yield) and residuals from the original fit are saved. The residuals are then increased by an adjustment factor (Stine 1990, p. 338), which is reported in the output file.

Bootstrapped data sets are then constructed by combining each saved predicted yield $\hat{Y}_{ij}$ with a randomly-chosen adjusted residual to arrive at a pseudo-yield value $\hat{Y}^*_{ij}$. (This procedure assumes that the statistical series weights w_i are correct.) The model is then refit, using the pseudo-yields in place of the original observed yields. The process is repeated (always using the original predicted values) up to 1,000 times.

From the bootstrap results, bias-corrected (BC) confidence intervals can be computed by standard methods (Efron and Gong 1983). The statistical literature recommends 1,000 bootstrap trials when computing 95% confidence intervals. ASPIC computes 80% confidence intervals, and should require fewer trials. The author recommends using at least 500 trials for bootstrap runs.

Some bootstrap trials may produce estimates (e.g., of K or MSY) outside the bounds set in the input file. Such trials are discarded automatically and replaced by new trials.

4.7 Input and output files

As noted, all ASPIC input and output files (Table 2 on p. 11) are in plain ASCII format. Sample files are provided with the ASPIC distribution.

4.7.1 Input file

An ASPIC input file contains all data and settings required for a single ASPIC run. It is recommended that when series of runs is made, that each input file be given a distinct name. This will ensure that the resulting output file names also are distinct.

The input file format is described in detail in §6 on page 13. The simplest way to generate an ASPIC input file is to run the command

```
aspic -help
```

from the command line, to generate the file `sample.inp`. That file can then be renamed and edited to the user's specifications. It may be useful to save an extra copy of the resulting file for use as a template.

4.7.2 Editing input files

To edit ASPIC input files and ASPICP control files, use a plain-text editor (programmer's editor) such as Windows Notepad. Good text editors are available as freeware, shareware, or commercial software.

A useful feature in any editor used for ASPIC input files is the ability to cut and paste rectangular blocks of text. A relatively simple editor with that feature is ConTEXT, which as of September 2009, was available free and locatable through Web search engines (e. g., Google). Other well known editors, such as emacs, are also suited to this task.

4.7.3 Output files

ASPIC produces several output files. Although all are written in plain ASCII, some are meant for human readability and others for use by ancillary programs. The main `.fit` and `.bot` output files, like

Table 2. Files read (R) or written (W) by ASPIC and related programs.

File type	Action	Used by	File contents and description
`.inp`	R	ASPIC	Input file with data, starting guesses, and run settings.
`.fit`‡	W	ASPIC	Main output file from FIT and IRF program modes.
`.bot`‡	W	ASPIC	Main output file from BOT program mode.
`.bio`†	W, R	ASPIC, ASPICP	Estimated B and F trajectory for each bootstrap trial; read by ASPICP (BOT program mode).
`.det`†	W	ASPIC	Estimates from each bootstrap trial (BOT program mode).
`.sum`	W	ASPIC	Optional file with summary of all runs made in a directory.
`.prn`	W	ASPIC	Estimated trajectories in a table easily read by S–Plus, R, SAS, or spreadsheet.
`.rdat`†	W	ASPIC	Detailed inputs and estimates specially formatted for reading by `dget()` function of S-Plus or R.
`.gen`	W	ASPIC	Summary results from GENGRID mode.
`.grd`	W	ASPIC	Summary results from LOGGRID mode.
`.ctl`	R	ASPICP	Control file in which user specifies projection parameters.
`.prj`‡	W	ASPICP	Projection results.
`.prb`	W	ASPICP	Extension to `.bio` file with projection results.

‡ File types readable by AGRAPH .

† File types intended for reading mainly by other computer programs.

all the files meant to be read by the user, have line lengths of 120 characters or less. Files meant for ancillary programs may have longer lines.

Names of output files are constructed from the root file name of the run's input file, with the appropriate file extension appended (Table 2).

· *Main output files.* The main ASPIC output file includes parameter estimates; measures of goodness of fit; and estimates of population benchmarks, biomass levels, and exploitation levels, as well as simple character plots. Output from bootstrap runs also includes bias-corrected confidence intervals on parameters and on other quantities of management interest.

The file extension of the main output file varies with mode of program operation (Table 2). Suppose the data (input) file is named `sword.inp`. Then in FIT and IRF modes, the main output would be written to a file named `sword.fit`. In BOT mode, the main output would be written to file `sword.bot`.

· *Bootstrap-related files.* Two additional files are generated after boostrap runs to allow further data analysis. The `.bio` file is used by ASPICP (described below) for projections following a bootstrap run. The `.det` file provides information on the individual bootstrap trials. It is not used directly by any supplied program, but is provided for the user's convenience.

· *Simple R or spreadsheet output.* (NOTE: The more complete `.rdat` file, described next, is recommended for R-compatible output.) A file (with extension `.prn`) containing a table of time-series data can be output from an ASPIC run in BOT or FIT program mode. It should be compatible with such programs as S-Plus, R, SAS, and spreadsheets. To enable this file, see specification of input file line 4 on p. 15.

· *More complete R-compatible output.* More complete output compatible with R is optionally available from ASPIC runs in FIT program mode. This file (extension `.rdat`), when read with the R function `dget`, becomes an R data object of type "list". For example,

```
sword <- dget("sword.rdat")
```

This author is periodically improving the `.rdat` output for completeness. For information on enabling this file, see specification of input file line 4 on p. 15.

· *Summary output from simulations.* To aid in simulation studies, a summary file (`.sum` file) can be written in the current directory. Enabling this file is described in the specification of input line 4 on page 15. The `.sum` file can be read by S-Plus or R with a statement like

```
myres <- read.table("aspic.sum",
   header = TRUE)
```

4.8 Starting ASPIC

First prepare an input file in the correct format (§6 on page 13). It's easiest to copy one of the sample input files provided to use as a template.

Then start the program, giving the input file name on the command line.[2] For example, the command

```
aspic sword.inp
```

or just

```
aspic sword
```

will cause the program to read an input file named `sword.inp` and produce corresponding estimates and output files. If only the command

```
aspic
```

is given, the program looks for the default input file, `ASPIC.INP`.

If the `.SUM` file as been enabled, summary output from each run in the directory will be written to it. The default name is `aspic.sum`. To use a different name for the `.SUM` file, give the name on the command line. For example, the command

```
aspic sword mysum
```

will read the file `sword.inp` and create (or write to) the summary file `mysum.sum`, along with the usual ASPIC output file(s).

Most errors detected while ASPIC is reading the input file will cause the program to print a descriptive message and stop. If the message is not clear, comparing the input file to the samples provided may reveal format errors.

[2] Most operations are also possible by dragging and dropping icons. The ASPIC Quick Reference, available from the ASPIC shortcuts folder installed on the user's Windows Desktop, includes more information on such use.

5 Overview of Auxiliary Programs

5.1 Overview of ASPICP

The auxiliary program ASPICP can be used following an ASPIC bootstrap run. It provides estimated time trajectories of population biomass and fishing mortality rate with bias-corrected confidence intervals. ASPICP is also used for making population projections beyond the observed data set. When making projections, the user can specify future harvests or effort levels, and the program projects biomass and fishing-mortality trajectories for up to 15 years past the original data. Printer plots of the trajectories are also provided.

ASPICP reads information recorded in the `BIO` file of the corresponding bootstrap run. The user controls the program with a simple control file, default `ASPICP.CTL`. Thus, the first step in using ASPICP is to create a proper control file with a text editor. Details of the file contents are given in §8, and sample ASPICP input files are included in the ASPIC distribution.

When starting ASPICP, the control file name is given on the command line; for example the command,

```
aspicp sword
```

or equivalently

```
aspicp sword.ctl
```

starts ASPICP as described in control file `sword.ctl`.

The output report from ASPICP is written to a `.PRJ` file whose name is derived from the `.CTL` file. As of ASPICP 3.16, detailed projection results from each bootstrap trial are written to a `.PRB` (projection `BIO`) file, essentially an extension of the `BIO` file (Table 2) that includes the projection years.

5.2 Overview of FTEST

A small program named FTEST is provided to perform significance tests when comparing different ASPIC models of a stock. The program is designed for comparing pairs of models that differ only in complexity (number of parameters). The TEST program has a text user interface. To run, type

```
ftest
```

at a Windows command prompt and answer the program's prompts.

5.3 Overview of AGRAPH

The Windows program AGRAPH is intended to provide quick, good-quality graphics of ASPIC and ASPICP results. Preformatted time-series plots of relative benchmarks and of observed and fitted abundance indices are provided. Plots can be viewed on the screen, sent to a Windows printer, or saved as graphics files in several formats.

The AGRAPH program was not meant to meet all graphics needs of ASPIC users. Instead, it allows one to examine results quickly and to have graphics suitable for assessment reports. Operation of AGRAPH is similar to that of any Windows program. It can also be started from the command line. For example, to make graphs from results in file `sword.fit`, use the command

```
agraph sword.fit
```

6 ASPIC Input File Specification

ASPIC reads its input from a single file containing control parameters and data. The format of that file is described here.

6.1 Generating a sample input file

A new feature of ASPIC 5.0 is that a sample input file can be generated with the command

```
aspic -help
```

The sample file is useful as a template for making new input files. It could also be helpful to have it available when reading this section.

6.2 General format guidelines

The representation of values in the input file must follow certain rules, which follow from the use of Fortran list-directed `read` statements to read the data.

- The exact position of values on a line is not important. However, if a line contains more than one value, they must be in the correct order.
- When a line contains more than one value, they must be separated by spaces (blanks). *Using* `tab` *characters to separate values is not recommended.*

The remaining rules depend upon the type of the data item (integer, real, or character).

- Each *real number* should contain a decimal point, an exponent (marked by the letter d or e), or a decimal point and an exponent. Examples: `1.0, 2e3, 1.3d6`. (Note that the notation `2e3` means 2×10^3.) An integer can be used in place of a whole real number.
- *Integers* must not contain decimal points or exponents. Examples: `0, 2, 94541`.
- *Character strings* may be delimited by matched apostrophes or quotation marks. However, this is necessary only if the string contains embedded blanks or other special characters. Examples (each on a separate line):
  `IRF`
  `'This is a valid string'`
  `"Another valid string"`
- Each line must have the specified number of values, separated by spaces. Values may not be otherwise arranged among lines.
- After the specified number of values have been read from a line, the program does not read it further. Thus, the rest of the line may be used to contain comments. Comments are included in the sample input files, preceded by pound signs, `##`. The pound signs are used to make the comments stand out to the eye and do not themselves denote comments to ASPIC.
- After all data have been read from the file, as determined by the number of years of data and number of data series, any further contents of the file are ignored by ASPIC. Thus, additional comments may be appended to the file.

6.3 The ASPIC input file, line by line

Line 1: Program mode

This is a character string of length 3, with possible values `FIT` (fitting mode), `BOT` (bootstrap mode), or `IRF` (iteratively reweighted fit mode). For further explanation of program modes, see §4.3 on page 8. For information on starting values in `BOT` mode, see §6.4 on page 18.

Line 2: Title of analysis

This is a character string of length 110 characters or less. The title is written to the main output file to identify the particular analysis. The title will also appear on graphs made with AGRAPH and projections made with ASPICP.

Since the title almost always contains spaces, it should be surrounded by quotation marks.

If the first character in the title is an asterisk (*), the main output file will contain control codes to activate the "lineprinter" font on many older laser printers. This provides a simple method of neatly printing ASPIC output files, which are 120 characters wide.

Example: `"*Run 4 for Redfin Tuna, 1994"`

An alternative for printing ASPIC output files is to use the freeware utility PrintFile: http://www.lerup.com/printfile/.

Line 3: Model shape and optimization control

✺ Note: In an effort to make ASPIC 5.0 as compatible as possible with earlier versions, the input file has the same general arrangement. However, additional control values are needed. Many of them appear on line 3, which make this section rather long.

❖ Line 3 has a varying number of items.

☞ The *first value* on line 3 is a character string specifying the model shape (program shape mode).

Value	Meaning
LOGISTIC	Fit the logistic (Schaefer) model.
GENGRID	Fit the generalized model at grid of values *or at one specified value.*
FOX	Fit the Fox model (a special case of GENFIT, below).
GENFIT	Fit the generalized model and estimate its exponent directly.

☞ The *second value* on line 3 is a character string specifying the conditioning mode for the fit. For more information, see §4.5.2 on page 9.

Value	Meaning
YLD	Condition fitting on yield (recommended for most analyses).
EFT	Condition fitting on fishing-effort rate.

☞ The *third value* on line 3 is a character string specifying the objective function.

Value	Objective function
SSE	Sum of squared errors (recommended default).
WTDSSE	SSE with annual data weighting.
LAV	Least absolute values (robust objective function).

Before setting values 4, 5, and 6 on line 3, define ϕ as the decimal fraction defining model shape, $\phi \equiv B_{MSY}/K$ (thus $0 < \phi < 1$). Then define $\Phi = \text{nint}(100\phi)$, where "nint" is the nearest integer function, and thus $0 < \Phi < 100$).

For example, in the logistic model, $B_{MSY} = K/2$, so $\phi = 0.5$ and $\Phi = 50$. For the Fox model, $\phi = \exp(-1) \approx 0.3679$ and $\Phi = 37$.

☞ The *fourth value* on line 3 is an integer, the lowest Φ to consider in GENGRID and GENFIT shape modes. A reasonable default might be 25. If present, this value is ignored in LOGISTIC and FOX shape modes.

☞ The *fifth value* on line 3 is an integer, the highest Φ to consider in GENGRID and GENFIT shape modes. a reasonable default might be 75. If present, this value is ignored in LOGISTIC and FOX shape modes.

☞ The *sixth value* on line 3 is an integer whose interpretation depends on the shape mode chosen:

Shape mode	Meaning of fifth value
GENGRID	Step size for grid of shape parameters examined.
GENFIT	Starting value for shape parameter.
LOGISTIC	Ignored.
FOX	Ignored.

In GENFIT shape mode, a reasonable default is often to use the logistic, i. e., to use 50.

☞ The *seventh value* on line 3 is a real number ξ that sets bounds to constrain the generalized fit near the logistic. For example, entering 8.0 here means that MSY for the generalized fit must lie between $1/8\times$ and $8\times$ the MSY estimated in the logistic fit. This ad hoc method is used to increase stability in fitting. In simulation studies, the value $\xi = 8.0$ has proven a reasonable default. The parameter K is constrained in the same way, but using $\xi' = 60\xi$.

During bootstrap runs, the change in bounds is effective only for the initial fit of the generalized model. For remaining bootstrap trials, bounds on MSY and K revert to those entered by the user on lines 19 and 20. *Thus, it is wise to adjust user guesses and bounds after fitting a point estimate and before fitting a bootstrap.*

If the input file specifies $\xi = 0.0$, no change of bounds is made. When $\xi \neq 0.0$, it is required that $\xi > 1.1$. As noted above, $\xi = 8.0$ is recommended as a general default.

To use a fixed model shape, use GENGRID *shape mode, set the fourth and fifth values equal to the specified shape, and set the sixth value (the step size) to zero.*

Examples of line 3

Example 1. Specify a grid-search for the shape parameter between $\Phi = 40$ and $\Phi = 60$ (a moderate range around the logistic, $\Phi = 50$). Use a step size in Φ of 5. Set bounds on MSY of $1/8\times$ to $8\times$ the logistic estimates. Use SSE (least squares) objective function (in log space), conditioned on (matching) the effort in the input file. Note that EFT here is the equivalent to the optimization mode CAT in ASPIC 3.x.

```
GENGRID EFT SSE 40 60 5 8.0
```

Example 2. Fit the generalized model conditioned on effort, with bounds of $\Phi = [40, 70]$ and starting value $\Phi = 50$. Constrain the generalized estimate closer to the logistic than in Example 1.

```
GENFIT EFT SSE 40 70 50 5.0
```

Example 3. Use a fitting procedure that could be accomplished with ASPIC 3.x. Fit the logistic model, conditioned on yield. When fitting the logistic model, only the first three values on line 3 are required.

```
LOGISTIC YLD SSE
```

Line 4: Verbosity & output file control

This is a single integer value that controls the amount of output printed to the screen during execution (the "verbosity") and whether the optional `*.sum`, `*.prn`, and `*.rdat` files (§4.7) are generated.

To control the amount of screen output, set the base value within the range 0–4. A setting of 0 gives almost no little screen output; 4 is intended for debugging, and gives too much for practical use. The recommended level is 2, moderate screen output.

To generate `.SUM` and `.PRN` files (recommended), add 10 to any of the above values. Thus, the recommended default would become 12.

To generate the `.RDAT` file, add 100 to any of the above values. The recommended default verbosity with all files generated would be 112. A value of 102 is also valid, to generate the `.RDAT` but not the other optional files.

The files are controlled in this way, instead of by a separate option in the input file, to maintain compatibility with older input files.

Line 5: Bootstrap control

Line 5 holds one or two integer values.

☞ The *first value* on line 5 is an integer $0 \leq n_1 \leq 1000$, which specifies the number of bootstrap trials. A reasonable default is 500. Although this value is used only in BOT program mode, it must be included in all input files.

☞ The *optional second value* on line 5 is an integer $30 \leq n_2 \leq 95$, which specifies user confidence interval levels (in BOT program mode). ASPIC provides two sets of confidence intervals on estimates. The first are 80% confidence intervals. The second by default are 50% intervals, but others can be specified by setting n_2. When specifying $n_2 > 80$, please set the number of bootstraps to 1000.

Line 6: Monte Carlo searching

This line contains two integers to control the optional Monte Carlo (MC) search during fitting.

☞ The *first value* on line 6 may be 0 to disable the Monte Carlo search during fitting; 1 to enable MC searching; or 2 for repeated searching. Turning MC on can help when a repeatable solution is otherwise difficult to find. Unfortunately, when strong local minima are present, the MC search can cause more problems than it solves. The author recommends leaving it off unless it is definitely needed.

☞ The *second value* on line 6 sets the initial number of Monte Carlo trials. when repeated searches

are enabled, this number is reduced by the program in searches after the first. Even if the first number on this line is 0, the second number is needed as a placeholder.

As stated above, the recommended procedure is to leave searching off unless it is needed. If turned on, suggested parameters are: `1 50000`. Monte Carlo searching, and particularly repeated searching, increases execution time considerably.

Line 7: Convergence criterion for optimizer

This convergence criterion is a real number denoted ε_1. After each adjustment of the simplex, the objective function is computed for each vertex of the simplex. Convergence is defined to occur when the following condition is met:

$$\frac{2\,|L_1 - L_0|}{L_1 + L_0} < \varepsilon_1.$$

where L_1 is the highest objective-function value in the simplex and L_0 is the lowest.

The recommended value is $\varepsilon_1 = 1 \times 10^{-8}$, which is written as `1d-8` in the input file. Using a different value is not recommended.

Line 8: Restart control

Randomized restarts are used by the ASPIC optimizer to avoid local minima. The two values (real, integer) on line 8 control this mechanism.

☞ The *first value* on line 8 is the tolerance ε_2 for ending restarts. When objective function values from k restarts in a row agree to within this tolerance, the solution is accepted. The recommended value is $\varepsilon_2 = 3\times10^{-8}$, which is written as `3d-8` in the input file. Changing this value is not recommended.

✺ ☞ The *second value* on line 8 sets k, the minimum number of restarts required. The recommended default for this integer value is 6. Larger numbers can be used if needed to obtain a solution not overly sensitive to starting values. (The value used by ASPIC 3.x was $k = 3$, increased in version 3.89 to $k = 6$.)

Line 9: Control of iterative computations

Iterative computations are used by ASPIC in several places. The two values (real, integer) on line 9 control two important sets of iterative computations.

☞ The *first value* on line 9 is the tolerance ε_3 for computing the annual fishing mortality rate (F). When conditioning on yield, an iterative method must be used to estimate F; it continues until successive estimates are within ε_3. The recommended value is $\varepsilon_3 = 1\times10^{-4}$, which is written as `1d-4` in the input file. Changing this value is not recommended.

In EFT optimization mode, ε_3 must be present in the input file, but it is ignored.

☞ The *second value* on line 9 is the number of time steps used per year for the generalized model, range 2–100. A reasonable default is between 12 and 24 steps. The choice affects execution speed (§2.2.1). ✺

When fitting the logistic (Schaefer) model, this number is not required. If present, it is ignored.

Line 10: Maximum estimated F

This line contains a real number specifying the maximum allowable estimate of F. This maximum (used when conditioning on yield) serves to aid the optimizer. The recommended default is `8d0`, which works well in most cases.

Line 11: Statistical weight for B_1 penalty in objective function

This line contains a real number that controls the influence of the penalty term on $B_1 > K$ (see §4.5.1). To omit the penalty term, set this to `0d0`. To use the penalty term, enter a positive real number (usually `1d0`). The penalty is useful in analyses showing a sharp decline in relative abundance in the initial years; such data sets can otherwise result in an extremely high estimate of B_1.

The recommended default is no penalty. If the resulting estimate of B_1/K is too high, the analyst can try either the penalty term or fixing B_1/K rather than estimating it. Either approach can affect estimates of management quantities; sensitivity analyses are useful to examine this. The penalty term is described in Prager (1994); fixing B_1, in Punt (1990).

Line 12: Number of data series

This line has a single integer from 1 to 10 that indicates how many data series are to be analyzed. The types of allowable data series are summarized in Table 1 on page 10.

Line 13: Series-specific statistical weights

The program reads as many real numbers from this line as series were specified on the preceding line. The statistical weight w_i for series i is multiplied by each squared residual for that series when the objective function is computed. When IRF program mode is used to analyze more than one data series, the w_i are adjusted to implement inverse-variance weighting. They can all be set to unity, best written 1d0 in the input file, unless there is reason to set them otherwise.

Line 14: Starting guess for B_1/K

This line contains a single real number between zero and one. Set this value based on your belief about the stock's condition at the start of the data set. In the absence of other information, a reasonable default is 0.5. See also §6.4 on page 18.

Line 15: Starting guess for MSY

In the absence of other information, half the largest yield can be used as a starting guess. This should be entered as a real number. See also §6.4 on page 18.

Line 16: Starting guess for K

In the absence of other information, a reasonable guess is 2 to 20 times the largest recorded yield. This should be entered as a real number. See also §6.4 on page 18.

Line 17: Starting guess(es) for q

The program reads as many real numbers from this line as there are data series specified on line 12. The meaning of q depends on the data type that it refers to. When it refers to an effort–yield data series (code CE in Table 1), q is the catchability coefficient. When it refers to a biomass index data series (codes I0, I1, or I2, Table 1), q is the constant relating the index data to the internal ASPIC estimates of biomass; e.g., if $q = 2.0$, the index data are divided by 2.0 before being compared to the estimated biomass. When it refers to a biomass estimate series (codes B0, B1, or B2, Table 1, the user's value of q is ignored, but a number must be present as a placeholder.

For technical reasons, optimization is more difficult when q is large. Thus, a successful result is most often obtained when the catch and index data are scaled so that all $q_i < 0.01$. This, of course, does not apply to Bn data series, for which by definition $q_i = 1$. See also §6.4 on page 18. ❖

Line 18: Flags to estimate (or fix) individual parameters

If line 12 specifies I data series, the program reads $I + 3$ integer values ("flags") from this line. The flags refer, in order, to B_1/K, MSY, K, and $q_i, i = \{1, 2, \ldots, I\}$. Set the flag to 1 to estimate the corresponding parameter, or 0 to keep the parameter constant at the starting guess. No flag should be set to any value other than 0 or 1. Although q_i is not estimated for some series types, $I + 3$ flags are always required.

Line 19: Bounds on MSY

This line contains maximum and minimum bounds on the estimate of MSY. These two real numbers are used to limit the solution to reasonable values. The user defines what is reasonable by setting these values. If final estimates are at either constraint, an error message is printed on screen and in the output file. Bootstrap trials falling outside these bounds are discarded. See also §6.4 on page 18.

Line 20: Bounds on K

This line contains maximum and minimum bounds on the estimate of K. They are used in the same way as the bounds on MSY. See also §6.4 on page 18.

Line 21: Random number seed

Use a large (7-digit) positive integer. Different numbers result in different random number sequences. Using the same seed allows duplication of a previous run.

Using a different seed should result in the same answer (within expected computation errors); if results are substantially different, at least one of the solutions was a local minimum. The user can attempt to remove sensitivity to random number seed by increasing the number of restarts required (second number on line 8).

LINE 22: NUMBER OF YEARS IN DATA SET

The total number y of years described by the input file, including any years with missing values. Within the file, each data series must be of length y and describe the same specific years. Nonoverlapping series can be accommodated by padding each series with missing values or zeroes as appropriate.

FOLLOWING LINES: INDIVIDUAL DATA SERIES

There must be one data block (group of lines) for each data series. Each block should include data for all y years; thus, thus each data block must be the same length y. The composition of each block is as follows —

(a) On the first line of the block, a series title (character string, length ≤ 40, in quotes). Example: `"Spring survey & total landings"`

(b) On the second line of the block, a character string of length 2 with the type code for the series. Type codes are listed in Table 1 (p. 10).

(c) Starting on the third line of the block, one data line for each year, with the following data on each line, separated by blanks —

(c1) *First number* — the year or other ID number. These are consecutive integers and must be identical from block (data series) to block. Numbers greater than `9999` will not print correctly.

(c2) *Second number* — a real number whose meaning depends on the series type. For type `CE`, it is the fishing-effort rate f for the year. For type `CC`, it is the average relative abundance (usually based on CPUE). For types `B0`, `B1`, or `B2`, it is a stock-biomass estimate. For types `I0`, `I1`, or `I2`, it is a relative abundance value.

(c3) *Third number* — a real number, required for `CE` or `CC` series only, giving the total yield (catch in biomass) from the fishery for that year. For other types of series (`B`n or `I`n), the third number is not needed.

(c4) *Final number* — an additional real number, used only when the objective function is set to `WTDSSE`. In that case, it is an annual statistical weight, which in each year is multiplied by the corresponding weight on line 13 to provide final statistical weighting for the observation.

Although yield–effort data series are designated type `CE`, effort is entered before yield on these lines. Similarly, in series type `CC`, the relative abundance appears before the yield. ❖

As noted in §6.3, it is recommended (although not absolutely necessary) to scale the catch and index data so that all $q_i < 0.01$. This does not apply to `Bn` data series, for which by definition $q_i = 1$.

6.4 Starting guesses and bounds

The user may benefit from changing starting guesses after using `FIT` mode, before using `BOT` mode. Thus, it is advisable to generate point estimates in `FIT` mode before using `BOT` mode. After examining the results from `FIT` mode, consider adjusting starting guesses and bounds on lines 14–17 and lines 19–20. Using better starting guesses and more restrictive bounds often results in time savings or better stability in fitting. However, bounds should be wide enough to encompass all plausible values.

6.5 Common questions about data series

6.5.1 MISSING VALUES AND ZEROES

Missing or zero data values are allowed in an ASPIC input file in some cases, depending on the conditioning mode and type of data series. All possible cases are described in Table 3, along with the action taken by ASPIC. A data line with a missing value or with $f = 0$ does not contribute to the objective function; however, the information present on the line is used in the analysis and does influence the estimates.

Any negative data item in the input file is considered a missing value by ASPIC. Thus a value can be set missing by inserting a minus sign in front of it, and the value can be restored in a later analysis by removing the minus sign. When a missing value appears in the ASPIC input file, an estimate of the underlying value appears in the output file.

Missing values are always distinct from true zero values. Zero should never be used to indicate a missing value, and a negative number should never be used for an observed zero.

Zero values of the abundance measure (CPUE) are never permitted, because it is assumed that the resource is not extinct during the analysis period. If an abundance index calculated prior to using

Table 3. Actions taken by ASPIC when data series include data record(s) with missing value(s) or zero(es). Dash (—) indicates normal data (neither missing nor zero). M *indicates a record with missing datum;* Z*, with zero datum. Series type* Index *includes* I0*,* I1*,* I2*,* B0*,* B1*, and* B2 *series.*

Cond. mode	Series type	CPUE or effort	Yield	Action by ASPIC
YLD	CC	M	—	Fit; estimate missing CPUE.
YLD	CC	—	M	Stop: missing yield not allowed when conditioning on yield.
YLD	CC	M	M	Stop: missing yield not allowed when conditioning on yield.
YLD	CC	M	Z	Fit with $F = Y = 0$ (no fishing).
YLD	CC	Z	M	Stop: missing yield not allowed when conditioning on yield.
YLD	CC	Z	—	Stop: zero CPUE never allowed.
YLD	CC	—	Z	Fit with $F = Y = 0$ (no fishing).
YLD	CC	Z	Z	Stop: zero CPUE never allowed.
YLD	CE	M	—	Fit; estimate missing CPUE.
YLD	CE	—	M	Stop: missing yield not allowed when conditioning on yield.
YLD	CE	M	M	Stop: missing yield not allowed when conditioning on yield.
YLD	CE	M	Z	Fit with $F = Y = 0$ (no fishing).
YLD	CE	Z	M	Stop: missing yield not allowed when conditioning on yield.
YLD	CE	Z	—	Stop for error: when $F = 0$, Y must be 0.
YLD	CE	—	Z	Stop: zero CPUE never allowed.
YLD	CE	Z	Z	Fit with $F = Y = 0$ (no fishing).
YLD	Index	M	—	Fit; estimate missing CPUE.
YLD	Index	Z	—	Stop: zero CPUE never allowed.
EFT	CC	M	—	Stop: missing CPUE not allowed when conditioning on effort.
EFT	CC	—	M	Stop: missing yield not estimable in this case.
EFT	CC	M	M	Stop: missing effort not allowed when conditioning on effort.
EFT	CC	M	Z	Stop: missing effort not allowed when conditioning on effort.
EFT	CC	Z	M	Stop: zero CPUE never allowed.
EFT	CC	Z	—	Stop: zero CPUE never allowed.
EFT	CC	—	Z	Fit with $F = Y = 0$ (no fishing).
EFT	CC	Z	Z	Stop: zero CPUE never allowed.
EFT	CE	M	—	Stop: missing effort not allowed when conditioning on effort.
EFT	CE	—	M	Fit; estimate missing catch
EFT	CE	M	M	Stop: missing effort not allowed when conditioning on effort.
EFT	CE	M	Z	Stop: missing effort not allowed when conditioning on effort.
EFT	CE	Z	M	Estimate with $F = Y = 0$ (no fishing).
EFT	CE	Z	—	Stop; if $F = 0$, Y must be 0.
EFT	CE	—	Z	Stop: zero CPUE never allowed.
EFT	CE	Z	Z	Estimate with $F = Y = 0$ (no fishing).
EFT	Index	M	—	Fit; estimate missing effort.
EFT	Index	Z	—	Stop: zero CPUE never allowed.

ASPIC is zero in a given year, one could try using a small number (e.g., 20% to 50% of the lowest nonzero value) in its place. Use of an *extremely* small number (e.g., 1% of the lowest nonzero value) will usually result in a large residual during the ASPIC analysis; such a residual can influence the results strongly. Thus, converting zeroes to very small numbers is not recommended.

6.5.2 Allocation of yield among series

When analyzing more than one data series, it is not always possible — or desirable — to associate a

yield with each measure of fishing effort rate or relative abundance. A common example is having several abundance indices for a stock, but only the total annual yield. This section aims to describe how ASPIC assumes yield is allocated among data series.

Yield is entered in both CE and CC series. Because ASPIC derives an abundance index from each CE series, it is important that the yield in a CE series correspond to the fishing-effort rate in the same series.

In contrast, it is not necessary for the abundance index in a CC series to correspond to the yield in the same series. For example, a valid CC series might have an abundance index computed from one fishery on a stock, paired with the total catch from all fisheries on that stock.

Despite the above, it is important that yield, summed across series, represent a constant proportion (usually assumed 1.0) of total removals. Changes in that proportion, whether due to reporting changes or changes in discarding practices, violate a fundamental assumption of ASPIC (and of most other assessment models). The consequences of that violation will depend on its severity.

❖ 6.5.3 Caution on zeros, missing values

The author has attempted to ensure that results of computations including missing and zero values are correct under all combinations of data series type, conditioning mode, and model shape. To that end, a simple test has been done of every combination shown in Table 3. Still, some cases occur infrequently in real data and so have not been tested repeatedly. Users are urged to examine results critically when missing and zero values are used and to advise the author if any problems should arise.

7 Advanced Options for ASPIC

As of version 5.34 of ASPIC, using an `aspic.ini` file to control advanced options is no longer supported. Several output files can be turned on and off with the verbosity setting (line 4 of input file), as described on page 15. User confidence intervals are specified on line 5 of the input file—see page 15.

8 ASPICP Input File Specification

The control file for ASPICP is relatively short; it should have the file extension `.CTL`. For the correct way to represent different data types in the file, see §6.2. A sample file is provided with the ASPIC distribution.

The following file format will work with all ASPICP versions. An expanded input file format is available for use only with version 4.0 and up. The new format allows added stochasticity in the projection and several additional ways of specifying management in the projection period. Documentation is not yet complete. Please see a version 4.0 `.CTL` file for an example.

8.1 Line by line

Line 1: Projection title

This is a character string, length ≤ 70. The title usually contains blanks and should be delimited by quotation marks. The ASPICP output file will also include the title of the original ASPIC run.

Line 2: Name of `.BIO` file

A character string specifying the name of a `.BIO` file from an ASPIC bootstrap run. Results from ASPICP will be based on the data in that file.

Line 3: Any character string ✺

In earlier versions, this line contained the output (`.PRJ`) file name. As of version 3.15 of ASPICP, the output file always has the same root file name as the input (`.CTL`) file and ends with file extension `.PRJ`.

Line 4: Confidence intervals ✺

In previous versions, a real number was required here as a placeholder, but was not used in the computations. That is still acceptable, and using any real number will cause defaults to be used for the options described immediately below.

As of version 3.16 of ASPICP, additional user options can be given on this line in the form of one character string and one integer:

☞ The *first value* on line 4 is the type of confidence interval to use. This should be either BC for bias-corrected confidence intervals (Efron and Gong 1983), or PC for simple percentile confidence intervals. The latter may be useful when the BC intervals appear irregular.

☞ The *second value* on line 4 is an integer, either 1 to smooth the resulting confidence intervals or 0 to use unsmoothed intervals.

The default when the user options are not given (i. e., when a placeholder is given instead) is to compute BC intervals and smooth them.

The author is aware that some BC intervals from ASPICP are very wide in certain years. Because that seems illogical, the author has checked the corresponding programming code repeatedly, but has not discovered any errors. Using PC intervals has avoided the irregularity in the cases tested.

LINE 5: NUMBER OF YEARS TO SKIP AT START OF PLOTS

An integer with recommended values 0 to 3. The first few years of biomass and mortality estimates are especially imprecise. Also, analysis of certain data sets can give in very high estimated biomasses in the first few years. Thus, omitting the first few years from the plots can be useful.

LINE 6: NUMBER OF YEARS OF PROJECTIONS

Integer between 0 and 15. The longer projections extend, the more speculative they are. For that reason, early versions of ASPICP limited projections to only 10 years, but some users found that overly restrictive. Projections are theoretical constructs and are most useful when comparing management strategies, rather than as forecasts of the future.

FOLLOWING LINES: MANAGEMENT REGIME TO PROJECT

Each following line has data for one projection year; the number of lines should equal the number of years specified on line 6. On each line, enter a real number followed by a single character. The real number represents the yield or relative fishing mortality rate to be applied that year, and the character tells which type of value the number is. For example, line 7 of the .CTL file might read

```
1.456d3   Y
```

to indicate that in the first projection year, a yield of 1,456 units will be taken. Thus, lines ending in Y are used for making projections conditioned on quota (TAC) management measures.

As another example, line 8 of the .CTL file might read

```
0.85d0   F
```

to indicate that in the second projection year, the fishing effort rate will be 85% of the rate in the final year of the original data. Thus, lines ending in F are used for making projections based on proportional reductions in fishing mortality rate. This use of relative values allows F-based projections to be made with reasonable confidence even when the estimated fishing mortality in absolute terms is quite imprecise.

F lines and Y lines can be mixed in the .CTL file. That might be done, e. g., when yield in the first projection year is already known, and management in subsequent years is to be by control of fishing effort.

8.2 Sample ASPICP input file

```
"Case with Y02=Y01; F03 to F07 = F(MSY)"
test.bio
XX
BC  1
0
6
1200   Y
0.55   F
0.55   F
0.55   F
0.55   F
0.55   F
```

9 Interpretation of ASPIC Results

This section explains some features of ASPIC estimates, and reviews considerations important when using ASPIC. Prager (1994) and Prager et al. (1996) contain additional discussion. ❖

9.1 Precision of parameter estimates

Production models tend to estimate some quantities considerably more precisely than others. Among the quantities more precisely estimated are maximum sustainable yield (MSY), optimum effort (f_{MSY}), and relative levels of stock biomass and fishing mortality rate. Here, relative levels means the biomass level relative to the level at which MSY is attained or the level of fishing mortality relative to that at which MSY is attained.

To provide more precise estimates, then, it is often useful to divide the stock-size estimates provided by ASPIC by the corresponding estimate of stock size at MSY (B_{MSY}). Similarly, the estimates of fishing mortality rate F are divided by F_{MSY} to obtain relative estimates. In its output files, ASPIC provides such relative estimates. The relative estimates present a more precise picture of the condition of the stock, because in normalization, the estimate of q — which is usually imprecise — cancels out.

In contrast, absolute levels of stock biomass (and related quantities), which include uncertainty in the estimate of q, are usually estimated much less precisely. One cannot place nearly as much credence in the absolute estimates of stock size, F, or any quantities that depend upon them. Absolute estimates of B_t and F_t from ASPIC are provided for the modeler's information and are not intended for use as management guidelines.

When two or more data series are analyzed, estimated ratios of catchabilities are typically estimated more precise than estimates of each q. Also, K may be estimated imprecisely or inaccurately even when MSY and f_{MSY} are estimated well. Again, this reflects the difficulty of translating relative biomass changes to an absolute scale.

The starting biomass, estimated as B_1/K, may be considered a nuisance parameter, and its estimate is often imprecise. Punt (1990) recommended fixing $B_1/K = 1.0$ (rather than estimating it) for the Cape hake stock off southern Africa, but it is not clear that that approach is appropriate for every stock. A similar approach is taken in using the penalty term described in §4.5.1.

To stabilize estimates from a particular data set, it can be useful to fit the model with B_1/K fixed at a range of values. Although the resulting estimates of the biomass trajectory will of course diverge at the beginning, they may provide sufficiently consistent estimates of present stock status for management purposes.

9.2 Estimating several catchability coefficients

ASPIC can use more than one data series in estimation. The analyst should be aware that the underlying assumption is that each abundance measure reflects the entire stock, except for random error. Thus, using this feature is similar to deriving an abundance index from each series and averaging them together.

It is not recommended to use abundance indices that are uncorrelated or negatively correlated with one another, unless their overlap is short. When abundance indices present different pictures, CPUE might instead be standardized with a model to remove effects of vessel type, area, gear, season, etc., before fitting an assessment model. The resulting index of yearly abundance can then be used as a ŚCC' series with the total catch. This provides quicker and more reliable estimation from ASPIC, but more importantly, it removes explainable variation from the data, which would otherwise become noise.

Nonetheless, using several abundance measures directly in ASPIC (or any assessment model) allows examining the departure of each series from model predictions.

9.2.1 Catchability over time

The user can estimate separate catchability coefficients for different periods of time. This is accomplished in practice by putting the periods of time in separate data series, each padded with zeroes or missing values as appropriate. This procedure can be used to examine hypotheses about changing catchability with time, perhaps as a result of changing fishing gear or changing environmental conditions. In interpreting such models, there are several considerations.

One concern is estimating whether the improvement in fit obtained from a more complex model is statistically significant. An ASPIC model with time-varying catchability can be tested against the base model (i.e., the simpler model with constant catchability) with an F–ratio test. Here F is the F distribution of statistics, not fishing mortality rate. The test statistic is

$$F^* = \frac{(\mathrm{SSE}_s - \mathrm{SSE}_c)/\nu_1}{\mathrm{SSE}_c/\nu_2}, \tag{2}$$

where SSE_s and SSE_c are is the error sums of squares of the simple and complex models, respectively; ν_1 is the difference in number of estimated parameters between the two models; and ν_2 is the number of data points less the total number of estimated parameters. The significance probability of F^* can be obtained from standard tables of the F-distribution with ν_1 and ν_2 degrees of freedom.

A small program called FTEST is supplied with ASPIC to facilitate making certain such tests. This program assumes that the same data are used for both models, but are divided into different periods with different estimates of q. The weighting for the penalty term (line 11 in the ASPIC input file) should be set to zero for this F-ratio test to be theoretically correct. The FTEST program is interactive, and takes all input from the screen.

Three often-repeated caveats apply when using the F-ratio test for this purpose. First, hypothesis tests are invalid when suggested by examination of the data. Instead, the test should be suggested by external information, such as changes in gear. Second, the significance of a series of tests is less than that of a single test. For information on this point, consult a reference on multiple comparisons (e. g., Klockars and Sax 1986). Third, hypothesis tests generally assume correct specification of the model. Thus, significance probabilities of tests on assessment models are always approximate. This caveat is especially important when there is evidence that the model does not fit well.

A nonparametric test of the null hypothesis $q_1 = q_2$ can be conducted from the fitting results. This test is constructed by examining the bootstrap estimates of the ratio of the two catchability coefficients. As an example, assume that the alternative hypothesis is that $q1 \neq q2$. Then the null would be rejected at $P < 0.05$ if a bias-corrected 95% confidence interval on q_1/q_2 did not include the value 1.0. Like the F test, this test is approximate because of the possibility of specification error. In addition, bootstrapping residuals may underestimate the true variability present in a time series (Freedman and Peters 1984). This has been addressed to some degree in the current version of ASPIC by the adjustment made to the residuals before bootstrapping is begun.

9.3 Projections

The ASPIC suite includes the program ASPICP, which can be used to make projections of the stock's response to various management measures. In interpreting results from ASPICP, it is helpful to consider the differences between age-structured and non-age-structured models.

The author has participated in many assessments in which both age-aggregated production models (such as ASPIC) and age-structured (cohort) models were used on the same stock. Usually, estimates of relative population trend (B/B_{MSY}) and relative fishery trend (F/F_{MSY}) from the two models have been similar. Occasionally, when abundance indices are noninformative, the age-structured model can estimate population trends reliably, while the production model cannot. I have not yet encountered data that supported both types of estimation with reasonable precision, yet give strikingly different results from various models.

Nonetheless, one area in which from age-aggregated models tend to differ from age-structured models is in estimating recovery from overfishing. In the author's experience, projections based on production-model dynamics often estimate considerably faster recovery than do projections from age-structured methods. It is too early to say categorically that one type of projection is more accurate than the other, because few stocks so assessed have recovered from overfishing. Still, it is the author's opinion that results based on production models may be overly optimistic, particularly for species that mature at relatively old ages. Age-structured models can capture the lag in recovery that occurs as increased recruitment and survival propagate through the immature age classes. Non-age-structured models do not explicitly capture this, so in cases where depletion extends below the age of maturity, they may project recovery at a rate not attainable by the actual population.

9.4 Estimation difficulties

The information in this section is central to obtaining correct results. Please read it thoroughly.

The optimization method used in ASPIC (Nelder and Mead 1965) is quite robust, but in unmodified form frequently stops at local minima (these represent sub-optimal solutions). This has been addressed in ASPIC with a restarting algorithm that requires the same solution to be found several times in a row before it is accepted. In the author's experience, the resulting optimizer is reasonably effective at avoiding local minima.

Nonetheless, ASPIC, like other programs that attempt complex nonlinear optimization, occasionally finds local, rather than global, minima. Two features of the program—beyond the restarting algorithm already mentioned—are available to detect and remedy this problem. First, solutions obtained at local minima are often not reasonable, and this

will often cause one of the parameters to be estimated at either its minimum or maximum bound. In such a case, a warning message is printed, both on screen and in the output file.

A second feature that can help avoid local minima is an optional Monte Carlo phase of estimation. When enabled, this tries to improve the initial fit by randomly searching for a better one in the neighborhood of the initial fit. If multiple searches are enabled, a shorter Monte Carlo search takes place periodically during fitting. Although such searches considerably increase the time required to find a solution, they can be helpful in avoiding local minima. If a solution is difficult to find, it can be helpful to enable the Monte Carlo searches.

When fitting difficult data sets, it can be useful to make several runs with different random number seeds. Agreement among a number of runs suggests that the solution is stable.

Occasionally ASPIC fails to converge to a minimum at all. This often indicates that the data do not fit the model very well, which can sometimes be verified by examining the results with AGRAPH. When there is no fit, the input file should be checked for errors (e. g., reversed catch and effort values). Rarely, changing the maximum value of F allowed (line 10 of the ASPIC input file) can improve convergence, if the problem occurs in EFF optimization mode. If the objective function appears (from the screen output) to have been near convergence, simply trying a second ASPIC run that uses the first run's results as starting guesses can sometimes provide a good solution. If the model includes several data sets (fisheries), it can be useful to eliminate one or more of them, at least temporarily, to see if convergence can be achieved.

If none of these suggestions is successful, estimates can often be made with the following strategy. Set one parameter (usually B_1/K) to a fixed value by setting the corresponding estimation flag (line 18 of the ASPIC input file) to zero. A solution might be possible conditional upon that value of B_1/K. If this technique leads to a solution, a range of fixed values of B_1/K can be tried and the solutions examined. Similar values of the objective function among solutions indicate that the solutions are nearly equivalent in terms of fit. Although the solutions will differ somewhat, they still may be useful, especially as confirmatory information or if little other information is available for management.

Although ASPIC has been tested on thousands of simulated and real data sets and is believed to operate correctly, errors can exist in any computer programs. Any user experiencing bugs or suspected bugs is asked to send the author copies of the input and output files by email: Mike.Prager@noaa.gov. The author attempts to correct all errors promptly.

10 Program Change History

Changes beyond Version 5.00

- Version 5.01: Added error message to output for q out of bounds.
- Version 5.02: Increased internal file name length to 128 characters. This has become an issue for the new drag-and-drop version, ASPICPW.EXE.
- Version 5.03: Fixed rare problem in initializing simplex when no starting values of K, MSY, or q given by user. Also, increased minimum number of convergences required in LAV optimization mode from 5 to 8.
- Version 5.04: Fixed some inconsistencies and errors in handling missing values. Verified correct action in all 36 possible combinations of series type, missing and zeroes. See Table 3.
- Version 5.05: Added LRP and CV(LRP) to bootstrap output for use with REPAST.
- Version 5.06: Further increased length of filename variables.
- Version 5.07: Improved output to PRN file. Was not printing series > 1 correctly.
- Version 5.08: Further increased length of filename variables.
- Version 5.09: Changed FIT to correct situation with MC search mode 2 (repeated search). Previously, in initial fit of a bootstrap run, MC repeats were fewer than for same data in in FIT program mode.
- Version 5.10: Eliminated printing of AIC and F-test for Fox model. Added printing of revised bounds on MSY and K during generalized fit. Changed logic to restore the user's bounds on MSY and K before bootstrapping a generalized fit.

- Version 5.11: Revised ASPICP to version 3.16, which allows either BC or PC confidence intervals and writes a .PRB file. Revised manual to explain those changes.
- Version 5.12: Added option for .RDAT file.
- Version 5.13: Fixed bug in which some biomass-index series were handled wrong and estimation failed completely.
- For further changes, please see change log supplied in PDF format.

11 Source Code

The Fortran source code for this software uses certain proprietary routines from the book *Numerical Recipes* by Press et al., and for that reason can not be freely distributed. Numerical Recipes Software has kindly granted their permission (ID number V95038) for the author to supply the source code to users upon specific request. However, any source code so supplied must not then be redistributed to others.

The author also wishes to be aware of all distribution of the source code, so that any useful modifications or error corrections can be made in the master copy of the software to benefit all users.

If you require a copy of the Fortran source code for this software, please request it from the author. In your letter or email, please include the following:

1. Your true name, institutional affiliation, physical address, and email address or telephone number
2. Your agreement that you will not redistribute the source code to others.
3. Your agreement that, if you modify the source code, you will not distribute any resulting program (or programs), nor the modified source code, beyond your immediate working group at your own location.
4. Your agreement that, if you modify the source code, you will ensure that your users do not redistribute either the modified source code or any resulting program or programs.
5. Your agreement that if you identify errors in the software, you will contact the author promptly so that the errors can be fixed for all users.

References

Efron, B. E., and G. Gong. 1983. A leisurely look at the bootstrap, the jackknife, and cross-validation. American Statistician 47: 36–48.

Fletcher, R. I. 1978. On the restructuring of the Pella–Tomlinson system. Fishery Bulletin 76: 515–521.

Fox, W. W. Jr. 1970. An exponential yield model for optimizing exploited fish populations. Transactions of the American Fisheries Society 99: 80–88.

Fox, W. W., Jr. 1975. Fitting the generalized stock production model by least-squares and equilibrium approximation. Fishery Bulletin 73: 23–37.

Graham, M. 1935. Modern theory of exploiting a fishery, and application to North Sea trawling. Journal du Conseil International pour l'Exploration de la Mer 10: 264–274.

Freedman, D. A., and S. C. Peters. 1984. Bootstrapping a regression equation: some empirical results. Journal of the American Statistical Association 79: 97–106.

Haddon, M. 2001. Modeling and quantitative methods in fisheries. Chapman & Hall, Boca Raton, Florida. 406 p.

Hilborn, R., and Walters, C. J. 1992. Quantitative fisheries stock assessment. Chapman and Hall, New York, NY. 570 p.

Klockars, A. J., and G. Sax. 1986. Multiple comparisons. Sage University, Quantitative Applications in the Social Sciences, Paper No. 07–61.

Mohn, R. K. 1980. Bias and error propagation in logistic production models. Canadian Journal of Fisheries and Aquatic Science 37: 1276–1283.

Nelder, J. A., and R. Mead. 1965. A simplex method for function minimization. Computer Journal 7: 308–313.

Pella, J. J. 1967. A study of methods to estimate the Schaefer model parameters with special reference to the yellowfin tuna fishery in the eastern tropical Pacific ocean. Dissertation, University of Washington, Seattle.

Pella, J. J., and Tomlinson, P. K. 1969. A generalized stock production model. Bulletin of the Inter-American Tropical Tuna Commission 13(3): 419–496.

Prager, M. H. 1992a. ASPIC: A Surplus-Production Model Incorporating Covariates. Collected Volume of Scientific Papers, International Commission for the Conservation of Atlantic Tunas (ICCAT) 28: 218–229.

Prager, M. H. 1992b. Recent developments in ex-

tending the ASPIC production model. ICCAT Working Document SCRS/92/127. 10 p.

Prager, M. H. 1994. A suite of extensions to a nonequilibrium surplus–production model. Fishery Bulletin 92: 374–389.

Prager, M. H. 2002. Comparison of logistic and generalized surplus-production models applied to swordfish, *Xiphias gladius*, in the north Atlantic Ocean. Fisheries Research 58: 41–57.

Prager, M. H., and C. P. Goodyear. 2001. Effects of mixed-metric data on production model estimation: simulation study of a blue-marlin-like stock. Transactions of the American Fisheries Society 130: 927–939.

Prager, M. H., C. P. Goodyear, and G. P. Scott. 1996. Application of a surplus production model to a swordfish-like simulated stock with time-changing gear selectivity. Transactions of the American Fisheries Society 125: 729–740.

Punt, A. E. 1990. Is $B_1 = K$ an appropriate assumption when applying an observation error production-model estimator to catch–effort data? S. African Journal of Marine Science 9: 249–259.

Punt, A. E., and R. Hilborn. 1996. Biomass dynamic models. User's manual. FAO Computerized Information Series (Fisheries), No. 10. Food and Agriculture Organization of the U.N., Rome. 63 p.

Quinn, T. J., and R. B. Deriso. 1999. Quantitative Fish Dynamics. Oxford University Press, New York. 542 p.

Schaefer, M. B. 1954. Some aspects of the dynamics of populations important to the management of the commercial marine fisheries. Bulletin of the Inter-American Tropical Tuna Commission 1(2): 27–56.

Schaefer, M. B. 1957. A study of the dynamics of the fishery for yellowfin tuna in the eastern tropical Pacific Ocean. Bulletin of the Inter-American Tropical Tuna Commission 2: 247–268.

Shertzer, K. W., and M. H. Prager. 2002. Least median of squares: a suitable objective function for stock assessment models? Canadian Journal of Fisheries and Aquatic Science 59: 1474–1481.

Stine, R. 1990. An introduction to bootstrap methods: examples and ideas. Pages 325–373 *in* J. Fox and J. S. Long (eds.), Modern methods of data analysis. Sage Publications, Newbury Park, California. 446 p.

Williams, E. H., and M. H. Prager. 2002. Comparison of two estimators for the generalized production model. Canadian Journal of Fisheries and Aquatic Science 59: 1533–1552.